Autonomous deployment of heterogeneous mobile sensors

N. Bartolini, T. Calamoneri
Sapienza, University of Rome, Italy
{bartolini,calamo}@di.uniroma1.it

T. La Porta
Pennsylvania State University, USA
tlp@cse.psu.edu

A. Massini, S. Silvestri
Sapienza, University of Rome, Italy
{massini,simone.silvestri}@di.uniroma1.it

***Abstract*—In this paper we address the problem of deploying heterogeneous mobile sensors over a target area. We show how traditional approaches designed for homogeneous networks fail when adopted in the heterogeneous operative setting.**

Unfortunately, network and device homogeneity is an unrealistic assumption in most practical deployments. In order to deal with realistic scenarios, we introduce VorLag, a generalization of the Voronoi approach based on Laguerre geometry. We theoretically prove the appropriateness of our approach to the management of heterogeneous networks. In addition we demonstrate that VorLag can be extended to deal with dynamically generated events or uneven energy depletion due to communications.

Furthermore, by means of simulations, we show that VorLag provides a very stable sensor behavior, with fast and guaranteed termination and moderate energy consumption. We also show that VorLag performs better than other methods based on virtual forces.

I. Introduction

The deployment of mobile sensors is attractive in many scenarios. Mobile sensors may be used for environmental monitoring to track the dispersion of pollutants, gas plumes or fires. They may also be used for public safety, for example to monitor the release of harmful agents as a result of an accident. In such scenarios it is difficult to achieve an exact sensor placement through manual means. Instead, sensors may be deployed somewhat randomly from a distance, and then reposition themselves to provide the required sensing coverage.

The potential of such applications has inspired a great deal of work on algorithms for deploying mobile sensors. Most of this work has addressed the deployment of homogeneous sensors to achieve a uniform coverage of a certain density in a specific Area of Interest (AoI). In this paper we address two more practical and challenging problems: (i) the deployment of heterogeneous sensors to achieve full coverage, and ii) the deployment of sensors to achieve coverage of varying density within an AoI. The first application accommodates sensors that may have different sensing ranges due to design or operating conditions, for example depleted battery supplies or damage to a transducer. The second application addresses the need for a higher density of sensing resources at a particular site where perhaps an event has been detected and requires more analysis.

We propose an algorithm which is based on a generalization of the Voronoi approach presented in [1]. We find that the original algorithm does not solve the problems of deployment with heterogeneous sensors or varying density over a field. We discover that in these scenarios, sensors do not move to cover the required area, but instead stop moving when they wrongly perceive that they have covered their maximum area. To solve this problem we introduce the notion of Laguerre distance into the Voronoi algorithm, and with some other modifications show that the resulting algorithm, which we call VorLag, is able to solve both deployment problems effectively. The primary additions to the original Voronoi algorithm are the use of the Laguerre distance and the redefinition of several algorithm parameters to ensure algorithm termination and improve convergence time.

We compare VorLag with another class of well accepted deployment algorithms based on virtual forces. We modify one particular virtual force algorithm so that it may operate in scenarios requiring the use of heterogeneous sensors or deployment of varying density. We find that VorLag has better performance and characteristics than the virtual force algorithm. Because the virtual force algorithm balances the distance between sensors, and does not specifically target sensor heterogeneity or a specific coverage density, it takes longer to converge, and under some circumstances is not guaranteed to converge at all. Also, we find that unlike VorLag, the virtual force algorithm requires offline tuning of several parameters which have a large impact on its performance; a priori knowledge of the density of deployed sensors is required for the proper tuning of these parameters which makes the algorithm impractical.

In summary, our contributions are:

- We identify the limitations and their root causes when using Voronoi-based algorithms with heterogeneous sensors and when deploying sensors with varying density;
- We generalize a previously proposed Voronoi-based algorithm with the notion of Laguerre distance to solve the problem of deploying heterogeneous mobile sensors;
- We extend the new algorithm to execute in environments in which a varying coverage density is required;
- We extend a virtual force based algorithm to operate in these two scenarios;
- We compare the performance of the VorLag and virtual force algorithm and determine the fundamental causes behind the limitations of the virtual force algorithm.

The VorLag algorithm is practical in that it provides very

978-1-4244-4634-6/09/$25.00 ©2009 IEEE

stable sensor behavior, with fast and guaranteed termination and moderate energy consumption. It does not require manual tuning or perfect knowledge of the operating conditions, and works properly if the sensor positioning is imprecise. The algorithm only requires loose synchronization and local communication. Because it converges quickly and does not require a priori knowledge of the deployment environment, it is also well suited for dynamic environments in which the sensing density requirements change over time.

The paper is organized as follows. Related work is presented in Section II. In Section III we motivate the problem and introduce some preliminary concepts. Section IV presents the algorithm VorLag. In Section V we address the problem of density driven deployment. Section VI summarizes a virtual force based algorithm that we use for performance comparisons whose results are shown in Section VII. Section VIII concludes the paper with some final remarks.

II. Related Work

Various approaches have been proposed to self-deploy mobile sensors. The virtual force approach (VFA) models the interactions among sensors as a combination of attractive and repulsive forces. As a result of these antagonist forces, the sensors spread throughout the environment. One of the first algorithms based on VFA was presented in [2].

Drawbacks of VFA include complex required tuning of several parameters and an oscillatory behavior of sensors. Possible improvements to decrease the oscillations include the introduction of dissipative forces [2], [3] or the definition of arbitrary thresholds as stopping conditions [4], [5]. The tuning of such thresholds is laborious and relies on an off-line configuration.

The virtual force model is also at the basis of several other proposals [6], [7], [8], [9]. Of these proposals we focus on the one presented in [6] under which a dynamically calculated constraint on the length of sensor movements prevents oscillations. This algorithm is described in more detail in Section VI and is referred in this paper as a benchmark for performance comparisons.

Techniques based on computational geometry model the deployment problem in terms of Voronoi diagrams or Delaunay triangulations. According to [1], each sensor iteratively calculates its Voronoi polygon, determines the existence of coverage holes and moves to a better position if necessary. This approach inspired our proposal and it can be obtained as an instance of our general approach, when sensor capabilities are homogeneous. A dual approach exploits Delaunay triangulation [10]. This approach does not guarantee oscillation avoidance if proper threshold parameters are not set.

Other approaches introduce techniques for sensor deployment in a different operative setting than we consider [11]. Finally, an approach based on the construction of a regular triangular lattice is proposed in [12].

Recent papers [13], [14] focus on static sensor deployment with variable density in order to mitigate the effects of the uneven energy depletion due to communication with a sink [15], [16]. The work [3] introduces a unified solution for sensor deployment and relocation to adapt the density to the proximity of events of interest. The work [17] addresses the problem of sensor heterogeneity specifically, but under assumptions on the network topology that are very restrictive with respect to those considered in this paper.

We conclude this section by pointing out that the idea of using generalized Voronoi diagrams is not new in the area of sensor networks [18]. Nevertheless, to the best of our knowledge, none of the previous work uses the approach proposed here to address the problem of heterogeneous mobile sensor deployment or deployment of varying density.

III. Motivation and preliminaries

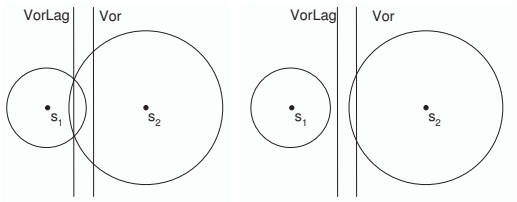
In the following subsections we show the limits of existing Voronoi based approaches to guide sensor movements and give the basics of our new algorithm.

A. Traditional Voronoi approach

A Voronoi diagram partitions the AoI into smaller polygonal regions so that each polygon is better covered by the sensor to which it belongs, rather than by any other one. The Voronoi algorithm is designed for homogeneous sensors. It provides that each sensor calculates its Voronoi polygon, and move towards its vertices if they are uncovered. Thus, each edge of the diagram lies on a line which partitions the AoI in two half planes, each one covered better by either one or the other of the two generating sensors. This property of the Voronoi diagram no longer holds in the case of heterogeneous sensors, but it is a necessary property for the diagram to guide sensor movements towards coverage holes, according to the Voronoi algorithm.

Figure 1(a) shows an example in which the line "Vor" is equidistant from the points s_1 and s_2. It is easy to see that since this line does not cross the intersection between the two circles, it does not partition the plane as required. Instead, the line labelled "VorLag" is drawn so as to assign each sensor to the half plane that it can cover best. Later in the paper we will show that this line is also equidistant from the points s_1 and s_2 but it is so in the Laguerre geometry. Figure 1(b) shows another case in which the Voronoi axis does not properly partition the AoI as it is done by the VorLag axis.

In order to show what kind of problems may occur when using the Voronoi algorithm in the case of heterogeneous sensors, we consider the example of Figure 2 in which sensors are deployed over a rectangular AoI, where the left zone is covered redundantly and the right zone is largely uncovered. The use of this algorithm produces no movements in such a configuration. The sensors with a small sensing circle (sensors 10-14) do not move, as their sensing range is completely included by their Voronoi polygon. The sensors with larger sensing circles (sensors 0-9), do not move either, because they already cover their own polygon completely. We can conclude that Figure 2 represents a critical configuration for the Voronoi algorithm for which the bad bisector placement leads to a *barrier effect*. This effect is one of the reasons of



(a) (b)

Fig. 1. Different positions of the line which is equidistant to s_1 and s_2 according to the Euclidean (Vor) and to the Laguerre distance (VorLag) in the case of (a) intersecting and (b) non-intersecting circles.

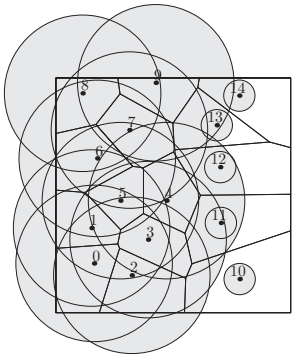


Fig. 2. Critical configuration for the traditional Voronoi algorithm.

the inapplicability of the traditional Voronoi approach even in the case of varying density requirements as will be more clear in Section V.

B. Voronoi diagrams in the Laguerre geometry

One of the most interesting generalizations of Voronoi diagrams concerns the notion of distance. The notion of Euclidean distance can be replaced by a variety of different formulations, among which we chose to use the one by Laguerre [19]. This choice is attractive as it preserves the property of generating straight edges, instead of complex curves, that is required for the Voronoi algorithm to work. The Laguerre-Voronoi diagrams are constituted by portions of straight lines which are perpendicular to the line segments connecting the centers of the associated generating points.

Given a circle $\mathcal{C}$ with center $C = (x_C, y_C)$ and radius r_C, and a point of the plane $S = (x_S, y_S) \in \Re^2$, the Laguerre distance $d_L(\mathcal{C}, S)$ between the circle $\mathcal{C}$ and the point S is defined in terms of the Euclidean distance $d_E(C, S)$ between points C and S: $d_L^2(\mathcal{C}, S) = d_E^2(C, S) - r_C^2$.

It should be noted that this metric is not a distance in the mathematical sense. Actually $d_L^2(\mathcal{C}, S)$ can be negative. The sign of $d_L^2(\mathcal{C}, S)$ depends on the position of S with respect to the circle $\mathcal{C}$. It is negative or positive if S lies inside or outside circle $\mathcal{C}$ respectively.

Lemma 1. *Given two circles $\mathcal{C}_1$ and $\mathcal{C}_2$ with non coincident centers C_1 and C_2, and radii r_1 and r_2, respectively, the locus of the points equally distant, in the Laguerre geometry, from the two circles is a straight line, called the radical axis of $\mathcal{C}_1$ and $\mathcal{C}_2$.*

The radical axis is perpendicular to the segment connecting the centers C_1 and C_2 of the two circles. This axis is located at distance k from C_1, with $k \triangleq \frac{d_E(C_1,C_2)}{2} + \frac{r_1^2 - r_2^2}{2 d_E(C_1,C_2)}$.

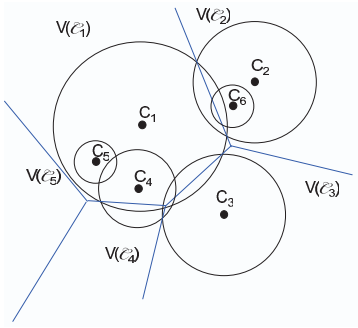


Fig. 3. Example of Voronoi-Laguerre diagram with empty and null polygons

The proof of Lemma 1 is an application of the Laguerre notion of distance, and it is omitted due to space limitations.

We now define the Voronoi-Laguerre diagram of n circles as follows. Given n circles in the plane $\mathcal{C}_i$ with centers $C_i = (x_i, y_i)$ and radii r_i, the *Voronoi-Laguerre polygon* $V(\mathcal{C}_i)$ for circle $\mathcal{C}_i$ is defined as

$$V(\mathcal{C}_i) = \{P \in \Re^2 | d_L^2(\mathcal{C}_i, P) \leq d_L^2(\mathcal{C}_j, P) \ \forall j \neq i\}.$$

Obviously, if $r_i = r_j$ for all $i, j \in \{1, 2, \cdots, N\}$, the Voronoi-Laguerre diagram reduces to the ordinary Voronoi diagram.

We highlight that the polygon $V(\mathcal{C}_i)$ may not contain any point of the plane. In this case, the Voronoi-Laguerre polygon $V(\mathcal{C}_i)$ is called a *null polygon*. We also note that the point C_i, that is the center of $\mathcal{C}_i$, may not lay inside $V(\mathcal{C}_i)$, even if this polygon is not null. A non-null Voronoi-Laguerre polygon $V(\mathcal{C}_i)$ that does not contain the center of its generating circle is called an *empty polygon*.

Figure 3 shows an example of Voronoi-Laguerre diagram that contains both null and empty polygons. Namely, the Voronoi-Laguerre polygon of the circle $\mathcal{C}_6$ is null, and the polygons of the circles $\mathcal{C}_4$ and $\mathcal{C}_5$ are empty. On the contrary the polygons of the circles $\mathcal{C}_1$, $\mathcal{C}_2$ and $\mathcal{C}_3$ are non-empty and consequently non-null.

The following theorem is fundamental for the formulation of the Voronoi-Laguerre algorithm that we will present in the next section.

Theorem III.1. *Given n circles $\mathcal{C}_i$ with centers $C_i = (x_i, y_i)$ and radii r_i, $i = 1, \ldots, n$, consider the Voronoi-Laguerre polygon $V(\mathcal{C}_i)$ for circle $\mathcal{C}_i$. The intersection of $V(\mathcal{C}_i)$ with all circles $\mathcal{C}_j$, with $i \neq j$, is contained in $\mathcal{C}_i$. In other words, it does not exist any point in $V(\mathcal{C}_i)$ contained in some $\mathcal{C}_j$, $j \neq i$ that is not also contained in $\mathcal{C}_i$.*

Proof: By contradiction, assume there exists a point $P \in V(\mathcal{C}_i)$ contained in $\mathcal{C}_j$ for some $j \neq i$ but not contained in $\mathcal{C}_i$. In view of the definition of Voronoi-Laguerre polygon, it

must be $d_L(\mathcal{C}_i, P) < d_L(\mathcal{C}_j, P)$ that is, equivalently,

$$d_E^2(\mathcal{C}_i, P) - r_i^2 < d_E^2(\mathcal{C}_j, P) - r_j^2. \quad (1)$$

On the other hand, If P is contained in $\mathcal{C}_j$ but not in $\mathcal{C}_i$, then $d_E(\mathcal{C}_j, P) \leq r_j$ and $d_E(\mathcal{C}_i, P) > r_i$. Substituting these inequalities in Equation 1 we get $0 < 0$, which is a contradiction. ■

Theorem III.1 motivates the important modifications we introduce to account for heterogeneous sensors. In the next section, we show how the concept of a Voronoi-Laguerre diagram can be applied to partition the AoI to support sensor movements.

IV. A Voronoi-Laguerre approach for heterogeneous sensor deployment

A. Assumptions

1) The sensing and communication radii of sensor i are r_i and r_i^{tx}, respectively, with $r_i \lessgtr r_j$, $r_i^{tx} \lessgtr r_j^{tx}$, $\forall i \neq j$.
2) The communication and the sensing radii of any two sensors are such that $r_i + r_j < \min\{r_i^{tx}, r_j^{tx}\}, \forall i \neq j$, that is any two sensors are able to communicate while their circles are touching.
3) Each sensor knows the coordinates of the AoI and can determine its own location (e.g. using low cost GPS, notice that the algorithm is not particularly sensitive to inexact location).
4) The sensors are loosely synchronized (in order to provide a round-based execution).
5) Sensors move at possibly different speeds v_i.

B. The idea

Our deployment algorithm, called VorLag, is based on the local calculation of the Voronoi-Laguerre diagram determined by the sensor locations over the AoI and their related sensing radii. Such a diagram partitions the AoI into disjoint polygons, each of them related to only one generating sensor. In view of Theorem III.1, if a sensor cannot detect a phenomenon in its polygon, no other sensor can detect it. Hence each sensor uses the information related to its Voronoi-Laguerre polygon to determine the presence of coverage holes and to decide future movements.

Our sensor deployment protocol runs iteratively. In each round, the sensors broadcast their locations and construct their local Voronoi-Laguerre polygons on the basis of the information received from neighbors. Each sensor evaluates the existence of coverage holes within its polygon and makes movements decisions accordingly.

It should be noted that in the Voronoi-Laguerre diagrams, there are some interesting situations that do not occur with ordinary Voronoi diagrams. First, the polygon of a sensor may possibly be null (the sensor has no polygon: this can happen when its sensing range is completely included in the union of the sensing circles of other sensors). In this case, the sensor should not move, as there is no direction that it can take to locally improve its coverage, and it should wait for other sensors to make place. Second, a sensor may be situated outside of its polygon, that is therefore necessarily empty. In this case, even if the polygon is completely covered, the sensor should move towards its polygon, as it is the portion of the AoI where it can contribute a better coverage. Even if the sensor covers its polygon completely, it can still improve the coverage regularity and uniformity by moving towards its own polygon.

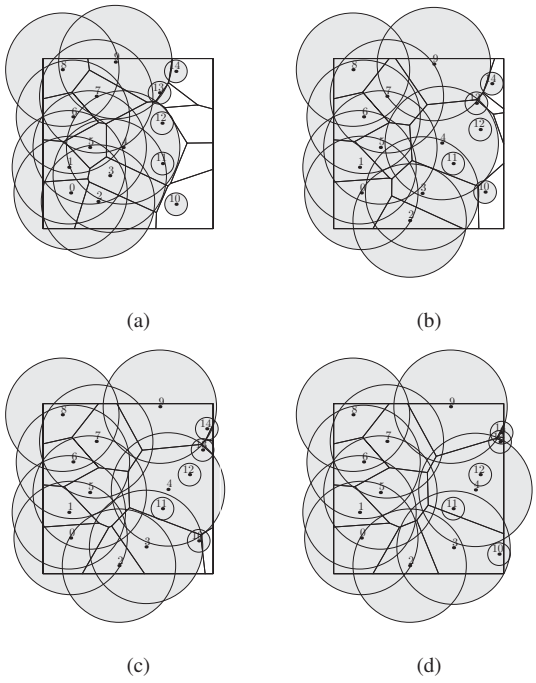


Fig. 4. Execution of the VorLag algorithm under an initial configuration that is critical for the traditional Voronoi approach: (a) initial configuration, (b) round 6, (c) round 9, (d) round 12 - final configuration.

In Figure 4 we show progressive runs of the algorithm execution under the initial configuration of Figure 2, that is not critical for VorLag, as there is no barrier effect.

C. The details

In this subsection we describe the algorithm in more detail: we first provide the description of the Voronoi-Laguerre diagram construction and of the way it guides the basic movements of the sensors, then we detail an algorithm optimization, namely movement adjustment.

To construct its Voronoi-Laguerre polygon, each sensor calculates the radical axes it generates with respect to all its neighbors, by using the locally available information. Each radical axis divides the plane into two half-planes, one of which must be eliminated; the intersection of all the remaining half-planes, is the Voronoi-Laguerre polygon of the sensor.

Notice that, due to the limited communication range, some nodes may construct the Voronoi polygon while out of communication range of their neighbors. Hence we account for some inaccuracies in the local construction of the polygons. This is true for the original Voronoi algorithm and remains true even for its generalization in the Laguerre geometry.

Notice that this polygon can be null, and when not null it can be empty. If a sensor has a null polygon, it does not move, as a null polygon is generated only when the sensor is in an overcrowded area. If the sensor has a non-null polygon, the algorithm determines the target location of the sensor s as the point inside its Voronoi-Laguerre polygon whose distance to the farthest Voronoi-Laguerre vertex is minimized. This point is called *miniMax* of the polygon of s, denoted as $O_m(s)$. Positioning sensors in their miniMax points can reduce the variance of the distances to the Voronoi-Laguerre vertices, resulting in a more regularly shaped Voronoi-Laguerre polygon, which better utilizes the sensors sensing capabilities.

We define $\vec{v}_{i,O_m(s_i)}$ as the vector directed from the position held by s_i to $O_m(s_i)$ and $d_{\mathtt{max}i}$ as the maximum distance that the sensor s_i is allowed to traverse in a single round along $\vec{v}_{i,O_m(s_i)}$. The introduction of $d_{\mathtt{max}i}$ is necessary to avoid an excessive movement in the case of inaccurate polygon construction due to the limited communication range. We set $d_{\mathtt{max}i} = \frac{r_{tx}}{2} - r_i$. It should be noted that $d_{\mathtt{max}i}$, together with the minimum sensor speed $v_{\mathtt{min}i}$, influence the duration of the algorithm round as it is determined by $d_{\mathtt{max}i}/v_{\mathtt{min}i}$, plus the communication phase during which neighbouring sensors exchange their positions.

In order to reduce useless movements and to ensure algorithm termination, we modify the *movement-adjustment* scheme used in [1] which states that before moving to its target location, a sensor checks whether the movement leads to an increase in the local coverage of its own polygon.

Our first change is to introduce an exception that concerns sensors having an empty polygon. These sensors continue to move until they reach a point that belongs to their Voronoi-Laguerre polygon, where they can contribute a better coverage.

Our second modification is to improve coverage close to the boundaries of the AoI. We will refer to this new scheme as *movement adjustment over the extended AoI*.

Let $r_{\mathtt{max}} = \max_{i=1\cdots N} r_i$ be the maximum among the sensing radii of the available sensors. We consider an *extended AoI*, AoI', by expanding the borders of the AoI with a band whose thickness is $r_{\mathtt{max}}$.

Each sensor calculates its coverage as the intersection of its sensing disk with its Voronoi-Laguerre polygon, by considering the AoI'. The movement adjustment scheme becomes an evaluation of the increase in the coverage of the region AoI'. Notice that this new formulation of the movement adjustment condition does not alter the calculation of the destination or the perceived presence of coverage holes by a sensor. Indeed, the miniMax and the coverage holes are determined by considering the polygon calculated over the AoI, whereas only the local coverage is calculated by considering the polygon over the AoI'.

It is easy to see that the new formulation of this condition permits a better coverage than by applying the original version of movements adjustment presented in [1].

Theorem IV.1. *The VorLag algorithm converges and terminates.*

Proof: The proof follows from the consideration that each sensor is allowed to move only if it increases the coverage of its local polygon. As all the polygons constitute a partition of the AoI', at each round the overall coverage of AoI' is either increasing or the algorithm terminates. Convergence follows from the boundedness of the AoI'. Details are omitted due to space limitation. ■

Theorem IV.2. *At any fixed round t, the complexity of the computation of each sensor s is $O(\Delta_t(s))$, where $\Delta_t(s)$ is the number of sensors that are in radio proximity to s.*

Proof: At each round t, sensor s discovers all $\Delta_t(s)$ sensors to which it is within radio proximity and computes its Voronoi polygon in the Laguerre measure. This takes $O(\Delta_t(s))$ time. Then, s has to compute the miniMax point of its Voronoi polygon in order to move towards it. Megiddo [20] proposed a solution for the equivalent smallest enclosing circle problem which is linear in the number of points that must be included in the circle. It follows that, applying this algorithm, the processing time of each sensor s at round t is $O(\Delta_t(s))$. ■

V. On the use of the Voronoi Laguerre Algorithm for density driven deployment

The algorithm described in Section IV finds a natural extension to the case of applications in which a deployment of homogeneous sensors is required at varying densities over the AoI. This scenario may be motivated by a variety of reasons. For example, as events occur in the AoI, it may be desirable to have a high density of sensors near the event. As new events appear or end, the density requirements will vary accordingly. A non uniform density deployment may also be desirable in a sensor network that centralizes communications towards a small number of sinks. This may cause an unbalanced energy depletion among sensors.

We translate the problem of how to deploy homogeneous sensors under heterogeneous density requirements into our previous problem of deploying heterogeneous sensors endowed with position dependent sensing radii under uniform coverage requirements. Observe that this translation can be applied even when sensors are heterogeneous.

To extend our algorithm to accommodate deployments at varying density we introduce two modifications: 1) we introduce the concept of *position dependent sensing radius*, 2) we introduce a new limitation to the maximum distance that each sensor can traverse at each round.

Let us assume that the heterogeneous density requirements are expressed by the function $\rho(x, y)$, where x and y are the coordinates over the AoI. To reach a deployment where the distance among the sensors depends on the desired density in the sensor locations, we define the position dependent sensing radius $r(x, y)$ as the radius of a regular hexagonal tessellation that would be obtained by optimally deploying the sensors at the desired density in the point of coordinates (x, y).

VorLag can be used in this new operative context with variable density if we consider each of the uniform sensors

as endowed with a position dependent sensing radius.

It should be noticed that the use of position dependent radius to modulate the deployment density would not be possible under the ordinary Voronoi algorithm because of the barrier effect shown in Figure 2.

As the sensing radius of sensor s_i varies with its own position, the value of $d_{\mathtt{max}i}$ no longer ensures that, at the destination, the sensor s_i will not overlap with the coverage disk of other sensors that are out of communication range with respect to its initial position. Hence, even the proper value of $d_{\mathtt{max}i}$ should be position dependent and calculated on the basis of the sensing radius that the sensor s_i would have at the movement destination (x', y'). We set $d_{\mathtt{max}i}(x', y') = \frac{r_{tx}}{2} - r(x', y')$. As the above value of $d_{\mathtt{max}i}(x', y')$ may be complex to calculate, depending on the specific density function, we refer to the following more conservative value, which does not depend on the particular sensor, nor on the destination coordinates: $d_{\mathtt{max}} = \frac{r_{tx}}{2} - max_{(x,y)\in AoI} r(x, y)$.

Notice that this is a conservative condition with repect to the one that would be imposed by using $d_{\mathtt{max}i}(x', y')$.

VI. On the use of the virtual force approach for the deployment of mobile heterogeneous sensors

In order to evaluate the performance of VorLag proposed in this paper, we compare it with an algorithm based on virtual forces called Parallel and Distributed Network Dynamics (PDND), proposed in [6]. In PDND the force exerted by s_i to sensor s_j is modelled as a piecewise linear function. Therefore it is repulsive when the distance between s_i and s_j is lower than an arbitrarily tuned parameter r^*; it is attractive when the distance is larger, until it vanishes at another arbitrarily set distance. The suggested formulation of this force respects the condition of Lipschitz continuity that is necessary to ensure the convergence of PDND. The single sensor movement is limited by a dynamically set upper bound that guarantees that the potential energy is always decreasing, hence avoiding oscillations.

PDND works under the assumption that sensors are homogeneous in terms of sensing and communication capabilities. In order to make the algorithm feasible for heterogeneous sensors, namely sensors having different sensing radii, we need to redefine the force that one sensor exerts on the others. According to the algorithm PDND, this implies the definition of the rest distance r^* at which the force exerted by two interacting sensors is null.

The setting of this parameter is critical, particularly when the number of sensors to be deployed is small. To fairly evaluate PDND, we propose two different settings for the value of r^*, both obtained as a combination of the sensing radii of two interacting sensors, r_i and r_j.

In the first case, to which we refer as PDND$_{\mathtt{SQRT}}$ we consider $r^* = \sqrt{3}(r_i + r_j)/2$. This choice is motivated by a generalization of the optimum sensor positioning in the case of sensors having identical radii r. In this case the distance between two sensors is $r^* = \sqrt{3}r$. This setting is not convenient when the number of sensors is scarce, because it generates extended overlapping areas when r_i and r_j are significantly different.

The second choice of r^*, to which we refer as PDND$_{\mathtt{SUM}}$, aims to minimize the overlaps, hence $r^* = r_i + r_j$; that is two sensors try to position themselves so that their sensing circles are tangential.

When the number of sensors is relatively small, e.g. insufficient to guarantee the coverage completeness, PDND$_{\mathtt{SUM}}$ provides a better coverage than obtained with PDND$_{\mathtt{SQRT}}$, whereas when the number of sensors is much larger than necessary, with PDND$_{\mathtt{SUM}}$ the forces exerted by sensors are higher than with PDND$_{\mathtt{SQRT}}$, thus causing more useless movements and delays in the algorithm convergence.

The fact that the two choices of r^* solve two distinct aspects of the problem of coverage highlights that parameter tuning is a critical issue for this algorithm. In fact, parameter tuning is a critical issue for all the algorithms based on virtual forces, as the inspiring model is inherently dynamic, and prone to oscillations that must be controlled by introducing many parameters.

The proposed modification of PDND that deals with the case of heterogeneous sensors, can be applied as well to the problem of deploying sensors at varying density, described in Section V. To this purpose we modify the expression of the force by substituting the position dependent sensing radius (discussed in Section V) into the formulas for r^* given by both the algorithm variants PDND$_{\mathtt{SUM}}$ and PDND$_{\mathtt{SQRT}}$.

VII. Experimental results

In order to evaluate the performance of VorLag and to compare it with the two versions of PDND described in Section VI, we developed a simulator on the basis of the wireless module of the OPNET environment [21].

We ran three sets of experiments. In the first set we study the deployment of heterogeneous sensors. In the second we consider varying density requirements within the AoI. In the last set, we consider time-varying density requirements due to dynamic missions.

We perform these experiments to quantify the differences in performance and behavior of VorLag and PDND caused by a fundamental difference in the algorithms: VorLag specifically targets a coverage density and then terminates, while PDND terminates only when its forces are balanced. We expect that this will result in VorLag converging to solutions faster than PDND and using less energy. In fact, in some instances, we expect that PDND will never converge while VorLag will always terminate.

In all of the following experiments, we consider a random deployment of sensors in an AoI of 80m $\times$ 80m. We set the communication radius of all sensors r_{tx} to 20m and the sensor movement speed to 1m/sec. For PDND we use the settings proposed in [6]: the length of a round is 1sec, and the minimum movement threshold is 0.1m. For VorLag, the duration of a round is determined by the distance $\max_i d_{\mathtt{max}i}$ and by the sensor speed. The plotted results are obtained by averaging the values of 20 simulation runs.

A. Heterogeneous sensors

In this set of experiments we consider a network composed of heterogeneous sensors. Hence, we set the radius of each sensor to a random value in the interval [1m,5m].

Notice that the termination of PDND is guaranteed as the definition of the force fulfills the requirement of Lipschitz continuity and we use a positive minimum movement threshold.

We compare the algorithms in terms of deployment performance (e.g. coverage and completion time) as well as energy metrics (e.g. average traversed distance and total consumed energy). In the experiments we progressively increase the number of deployed sensors from 100 to 600.

Figure 5(a) shows the percentage of area that is covered at the end of the deployment phase. As expected, the coverage increases with the number of sensors and is complete with a sufficient numbers of sensors with all the three algorithms. $\text{PDND}_{\text{SQRT}}$ achieves the lowest coverage because of large overlaps of the sensing circles, a consequence of the smaller distance at which sensors balance the forces.

As Figure 5(b) shows, all algorithms require a similar amount of time to reach the final coverage when fewer sensors are available. This is because when the number of deployed sensors is near or below the minimum required to achieve full coverage, virtually all available sensors must move, thus increasing the deployment time. As the number of sensors increases, VorLag requires significantly less time to achieve the final deployment, since the initial deployment is random and it terminates as soon as the area is completely covered. PDND requires more time because of the nature of the virtual forces: it is harder to find the balance of forces when there are more sensors available than necessary. For example, with 450 sensors, VorLag converges in about 40% of the time required by PDND.

Figure 5(c) shows the time needed to achieve 99% coverage of the AoI. We do not study the time to achieve full coverage because PDND does not achieve complete coverage. No point is depicted for fewer than 300 sensors since the available sensors are not sufficient to cover the entire area. VorLag achieves the desired coverage faster in all the considered scenarios. $\text{PDND}_{\text{SQRT}}$ requires more sensors than PDND_{SUM} and VorLag to achieve the same coverage. This is due to the shorter distance at which forces are balanced. Moreover $\text{PDND}_{\text{SQRT}}$ requires a longer time than PDND_{SUM} since the exerted forces have a smaller intensity, thus movements are shorter.

In Figure 5(d) we focus on a particular execution with 300 sensors. This figure reinforces the point of the previous results, showing that VorLag has a faster convergence to the same coverage. It not only terminates sooner, but it achieves a coverage of around 99% of the AoI in a few seconds of execution. These results highlight the capability of the VorLag algorithm to achieve satisfactory coverage in a very short time. This is a crucial property in the case of mission critical operating environments.

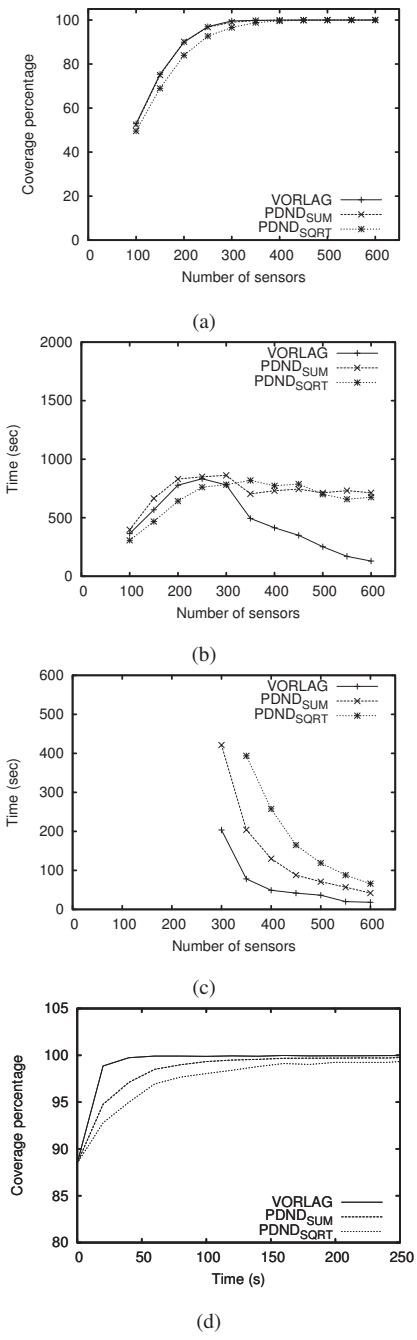


Fig. 5. (a) Percentage of covered AoI, (b) completion time, (c) coverage time and (d) Coverage over time.

We now compare the performance of the algorithms in terms of energy consumption. A sensor consumes energy as a consequence of movements (start/stop and traversed distance [22]) and communications (sending and receiving messages).

Figure 6(a) shows the average distance traversed by sensors. Since VorLag stops its movement as soon as the area is completely covered, the traversed distance decreases as the number of sensors increases. This is also true for PDND, although sensors traverse longer distances with respect to

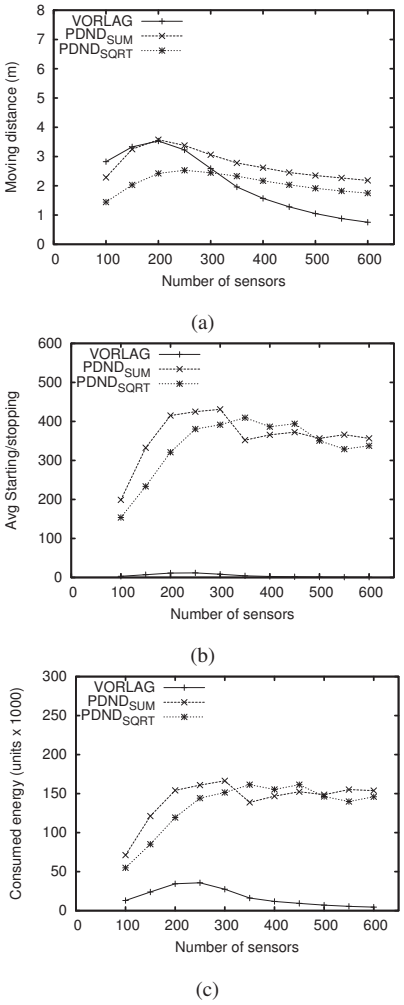


Fig. 6. (a) Average traversed distance, (b) average number of start/stop actions, (c) average consumed energy.

VorLag.

The average number of starting and stopping actions is shown in Figure 6(b). PDND performs a higher number of starting/stopping actions with respect to VorLag. This is due to the short distance that a sensor moves in each round. According to the definition of PDND, in order to guarantee the convergence of the algorithm, once a sensor has calculated the resulting force, it is allowed to move in the direction of the force for a short distance that guarantees a decrease of the potential energy in that direction. VorLag, on the other hand, makes longer (but still accurate) movements, resulting in a much lower number of starting stopping actions.

Figure 6(c) shows the unified energy consumption, which includes the energy spent by sensors for communications and movements, expressed in energy units. The reception of one message corresponds to one energy unit, a single transmission costs 1.125 units [23], a 1 meter movement costs 300 units and a starting/stopping action costs the same as 1 meter of movement.

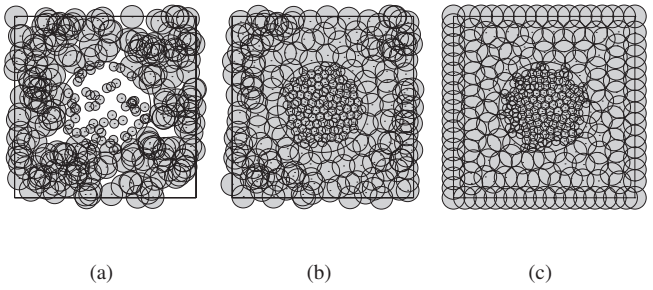


Fig. 7. An example of (a) random initial deployment of 300 sensors, (b) final deployment achieved by VorLag, (c) and PDND$_{\texttt{SUM}}$.

Because the energy spent for communication is generally much smaller than for movements, the energy consumption is dominated by the traversed distance and the number of starting/stopping actions. VorLag consumes less energy than PDND, whose performance is affected by the high number of starting and stopping actions. Moreover VorLag shows once again a good scalability with the number of sensors, as it requires less energy as the number of available sensors grow.

B. Density driven deployment

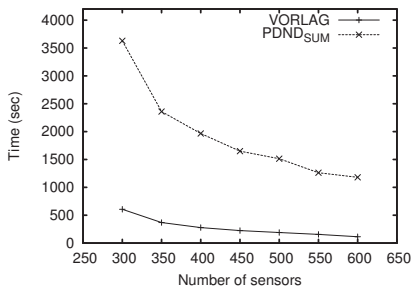
In the second set of experiments we consider the problem of covering an AoI with varying density. In these experiments, we assume that all sensors are homogeneous, i.e. have the same sensing radius r_s of 5m.

Figure 7(a) depicts a scenario with 300 sensors randomly placed over an AoI where a central disk requires more sensors than the rest of the area. The density requirement inside the central high density disk is 0.1 sensors per unit area, while the rest of the area only requires coverage. The smaller circles around the sensors located inside the higher density disk represent the higher density requirement, and therefore a lower position-dependent radius.

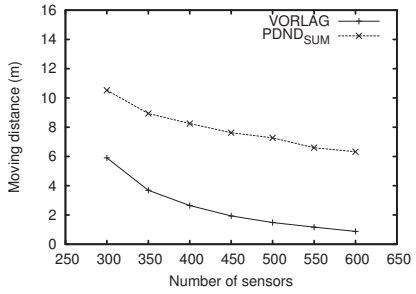
Figure 7(b) shows the final deployment achieved by VorLag (b), and by PDND$_{\texttt{SUM}}$. In these experiments we do not show the variant PDND$_{\texttt{SQRT}}$ because it performs similarly to PDND$_{\texttt{SUM}}$.

Note that, as the function representing the density requirements over the AoI is not continuous, the PDND force is not Lipschitz continuous, hence the algorithm convergence is not guaranteed. Setting a threshold on the minimum allowed movement under PDND will also not guarantee its termination because sensors do not stop moving as they do not reduce the overall potential energy with their movements. In order to have fair comparisons, we artificially terminate the execution of both algorithms as soon as they reach full coverage of the high density disk.

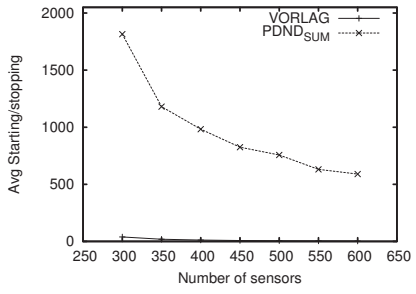
Figure 8(a) shows the completion time of the algorithms, i.e. the time required to reach the coverage of the denser disk at the required density level. VorLag outperforms PDND. In Figure 8(b) we show the average traversed distance during the deployment phase. VorLag shows a better behavior, allowing



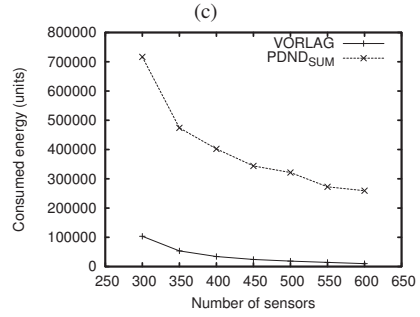
(a)



(b)

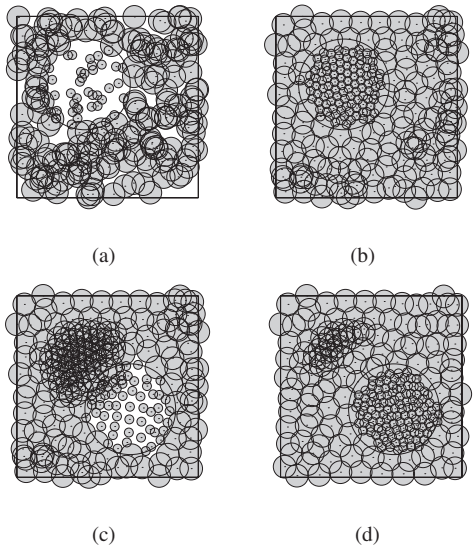


(c)



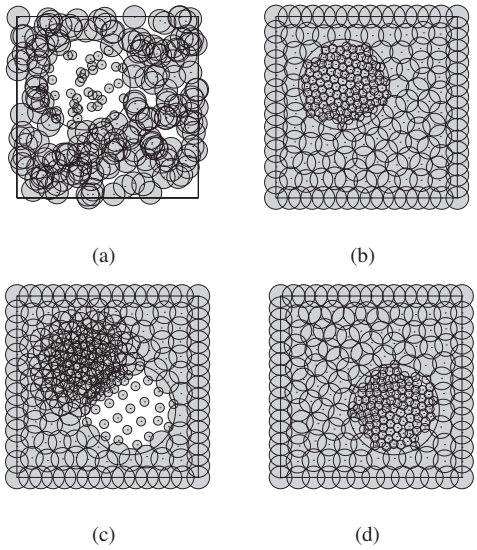
(d)

Fig. 8. (a) Completion time, (b) average traversed distance, (c) average number of starting/stopping, (d) average consumed energy.



(a) (b) (c) (d)

Fig. 9. Dynamic mission arrival: (a)-(d) execution of VorLag.



(a) (b) (c) (d)

Fig. 10. Dynamic mission arrival: (a)-(d) execution of PDND.

sensors to complete the deployment by traversing shorter distances with respect to PDND.

Figures 8(c) and 8(d) show the average number of starting/stopping actions and the average consumed energy, respectively. These results are explained in a similar way to those presented in Section VII-A.

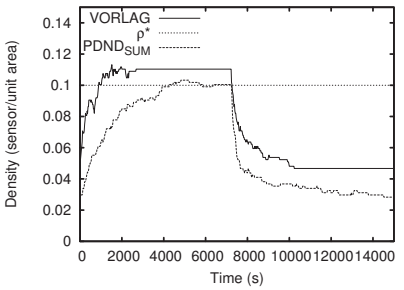
C. Dynamic mission arrival

In the last set of experiments we study the capability of the deployment algorithms to react to dynamically generated missions. A mission is seen as a change in the density requirements over the AoI. Such a change aims at making the sensor density high around the epicenter of the mission, in order to have a better detection of the ongoing event. The density of sensors around a mission is set to 0.1 sensors per unit area, as in the previous experiments.

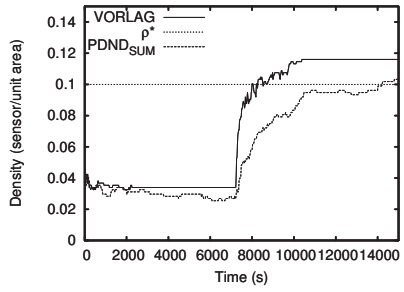
In these experiments we focus on the algorithm self-reconfiguration capabilities, i.e. on the time to relocate sensors in order to fulfill dynamic density requirements.

Figure 9(a) shows the initial deployment with 300 randomly placed sensors.

The disk located at top left of the AoI represents a first mission, known at the beginning of the deployment.



(a)



(b)

Fig. 11. (a) Density over time of the first mission area and (b) the second mission area.

Figures 9(b) and 10(b) show the deployment achieved by VorLag and PDND for the first mission, respectively.

Once the first mission has ended a second one appears at time t=7200 seconds, with the same density requirement, but located on the bottom right corner of the AoI, as depicted in Figure 9(c) and 10(c), and requires the network to reconfigure. Figure 9(d) and 10(d) show the final deployments achieved by VorLag and PDND respectively.

In Figures 11(a) and 11(b) we show the density over time of the first and second mission area respectively. VorLag shows a shorter configuration and reconfiguration time with respect to PDND, although PDND shows a more regular deployment. Note that in this context the goal of the algorithm is to achieve a minimum required density on the mission' locations and simple coverage elsewhere. Once this is achieved, there is no reason to move sensors in order to make the deployment more regular. The VorLag algorithm achieves its goal faster than PDND and does terminate, whereas PDND wastes energy with useless movements whose side effect is the creation of a more regular deployment pattern.

VIII. Conclusions

We introduce major modifications to the Voronoi based approach to the problem of deploying mobile sensors over an AoI. Unlike its original version, our approach works even in the case of heterogeneous sensors, and in the case of density requirements that vary over time and space. The proposed algorithm is self-adaptive to runtime changes of the operative conditions. It requires no prior knowledge of the applicative scenario, and does not necessitate manual tuning of its working parameters. We compared this algorithm with its traditional counterpart, showing that it resolves stale and critical situations. Furthermore we introduced modifications to a virtual force based approach in order to make fair performance comparisons. Extensive simulations show that our algorithm outperforms the other approaches in many respects.

References

[1] G. Wang, G. Cao, and T. L. Porta, "Movement-assisted sensor deployment," *IEEE Trans. on Mobile Computing*, vol. 6, pp. 640–652, 2006.

[2] A. Howard, M. J. Mataric, and G. S. Sukhatme, "Mobile sensor network deployment using potential fields: A distributed, scalable solution to the area coverage problem," *Proc. of DARS*, pp. 299–308, 2002.

[3] M. Garetto, M. Gribaudo, C.-F. Chiasserini, and E. Leonardi, "A distributed sensor relocation scheme for environmental control," *The ACM/IEEE Proc. of MASS*, pp. 1–10, 2007.

[4] J. Chen, S. Li, and Y. Sun, "Novel deployment schemes for mobile sensor networks," *Sensors*, vol. 7, pp. 2907–2919, 2007.

[5] N. Heo and P. Varshney, "Energy-efficient deployment of intelligent mobile sensor networks," *IEEE Trans. on Systems, Man and Cybernetics*, vol. 35, pp. 78–92, 2005.

[6] K. Ma, Y. Zhang, and W. Trappe, "Managing the mobility of a mobile sensor network using network dynamics," *IEEE Trans. on Parallel and Distributed Systems*, vol. 19, no. 1, pp. 106–120, 2008.

[7] Y. Zou and K. Chakrabarty, "Sensor deployment and target localization in distributed sensor networks," *ACM Trans. on Embedded Computing Systems*, vol. 3, no. 1, pp. 61–91, February 2003.

[8] S. Poduri and G. S. Sukhatme, "Constrained coverage for mobile sensor networks," *Proc. of IEEE ICRA*, pp. 165–171, 2004.

[9] M. R. Pac, A. M. Erkmen, and I. Erkmen, "Scalable self-deployment of mobile sensor networks; a fluid dynamics approach," *Proc. of IEEE IROS*, pp. 1446–1451, 2006.

[10] M. Ma and Y. Yang, "Adaptive triangular deployment algorithm for unattended mobile sensor networks," *IEEE Trans. on Computers*, vol. 56, pp. 946–847, 2007.

[11] G. Tan, S. A. Jarvis, and A.-M. Kermarrec, "Connectivity-guaranteed and obstacle-adaptive deployment schemes for mobile sensor networks," *The IEEE Proc. of ICDCS*, pp. 429–437, 2008.

[12] N. Bartolini, T. Calamoneri, E. Fusco, A. Massini, and S. Silvestri, "Autonomous deployment of self-organizing mobile sensors for a complete coverage," *Proc. of IEEE IWSOS*, pp. 194–205, 2008.

[13] X. Wu, G. Chen, and S. K. Das, "On the energy hole problem of nonuniform node distribution in wireless sensor networks," *Proc. of IEEE MASS*, pp. 180–187, 2006.

[14] M. Cardei, Y. Yang, and J. Wu, "Non-uniform sensor deployment in mobile wireless sensor networks," *Proc. of WoWMoM*, pp. 1–8, 2008.

[15] J. Li and P. Mohapatra, "Analytical modeling and mitigation techniques for the energy hole problem in sensor networks," *Pervasive and Mobile Computing*, no. 3, pp. 233–254, 2007.

[16] S. Olariu and I. Stojmenovic, "Design guidelines for maximizing lifetime and avoiding energy holes in sensor networks with uniform distribution and uniform reporting," *Proceedings of INFOCOM*, pp. 1–12, 2006.

[17] M. Lam and Y. Liu, "Two distributed algorithms for heterogeneous sensor network deployment towards maximum coverage," *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3296–3301, 2008.

[18] H. Ammari and S. Das, "Promoting heterogeneity, mobility, and energy-aware voronoi diagram in wireless sensor networks," *IEEE Trans. on Parallel and Distributed Systems*, vol. 19, no. 7, pp. 995–1008, 2008.

[19] H. Imai, M. Iri, and K. Murota., "Voronoi diagram in the Laguerre geometry and its applications," *SIAM J. Comput.*, vol. 14, no. 1, pp. 93–105, 1985.

[20] N. Megiddo, "Linear-time algorithms for linear programming in $\Re^3$ and related problems," *SIAM J. Comput.*, vol. 12, pp. 759–776, 1983.

[21] "Opnet technologies inc." *http://www.opnet.com*.

[22] G. Sibley, M. Rahimi, and G. Sukhatme, "Mobile robot platform for large-scale sensor networks," *IEEE International Conference on Robotics 158 and Automation (ICRA)*, pp. 1143–1148, 2002.

[23] G. Anastasi, M. Conti, A. Falchi, E. Gregori, and A. Passarella, "Performance measurements of mote sensor networks," *Proc. of ACM MSWiM 2004*, pp. 174–181.